Carnegie Mellon University
Software Engineering Institute

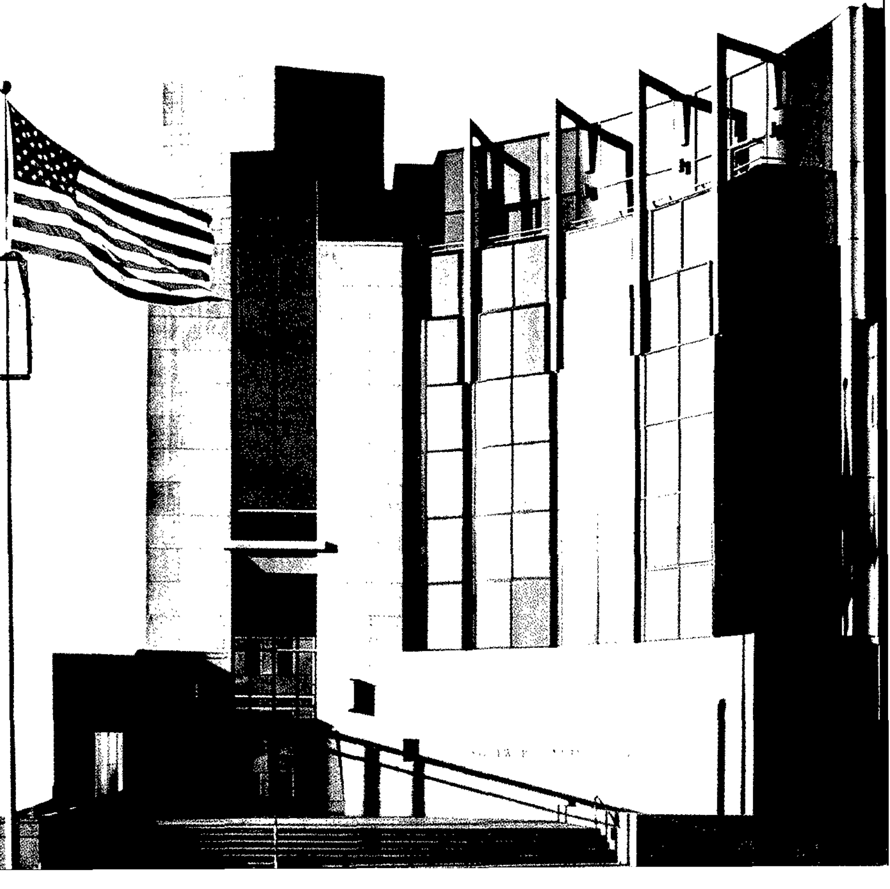
Enterprise Framework for the Disciplined Evolution of Legacy Systems

John K. Bergey
Linda M. Northrop
Dennis B. Smith

October 1997

TR

TECHNICAL REPORT
CMU/SEI-97-TR-007
ESC-TR-97-007



Technical Report
CMU/SEI-97-TR-007
ESC-TR-97-007
October 1997

Enterprise Framework for the Disciplined Evolution of Legacy Systems

John K. Bergey

Linda M. Northrop

Dennis B. Smith

Reengineering
Product Line Practice

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Table of Contents

List of Figures

Acknowledgments

We wish to thank Archie Andrews, Lisa Brownsword, Paul Clements, Wolf Goethert, Scott Tilley, and Nelson Weiderman of the Software Engineering Institute for providing comments on an earlier version of this paper.

This work was supported in part by the U.S. Department of Defense.

Enterprise Framework for the Disciplined Evolution of Legacy Systems

Abstract: Many organizations are planning to "migrate" their legacy systems to distributed open system environments or a single product line of systems. Many of these efforts are often less than successful because they concentrate on a narrow set of software issues without fully considering a broader set of enterprise-wide management and technical issues. This report describes an enterprise framework that characterizes the global environment in which system evolution takes place and provides insight into the activities, processes, and work products that shape the disciplined evolution of legacy systems. Exemplary checklists are included to identify critical enterprise issues corresponding to each of the framework's elements. Preliminary results indicate that the enterprise model is a useful tool for probing and evaluating planned and ongoing system evolution initiatives. The model serves to draw out important global issues early in the planning cycle and provides insight for developing a synergistic set of management and technical practices to achieve a disciplined approach to system evolution.

1. Introduction

Organizations everywhere are experiencing tremendous pressure to evolve their systems so they can better respond to marketplace needs and rapidly changing technologies. This constant pressure to evolve is driven by escalating customer expectations and the need to respond to new enterprise standards, incorporate new products and system features, improve performance, cope with endless new software releases, and stave off hardware and software obsolescence.

To effectively evolve legacy systems in this fast-paced environment, managers require answers to the following types of questions:

- How can we systematically sort out all the issues with which we are confronted?
- How do we plan the evolution of a large and complex system, including reengineering the system?
- What are the critical success factors of system evolution?
- How can we determine if we are on the right track?
- How do we evolve the system without adversely affecting operations?

Because there is not an established discipline of software evolution, organizations often focus on specific subsets of technical issues. In reality, however, many aspects of system evolution are not strictly a matter of addressing technical problems. For example, there is a tendency to focus on a narrow set of technical issues without considering the broader systems engineering issues, the increased needs of the customer, the strategic goals and objectives of the organization, and the business operations of the enterprise. A case in point is the "year 2000" (Y2K) crisis [Smith 97]. This crisis appears to be a simple, well-bounded, technical problem, but in reality must be addressed from an enterprise perspective that accounts for the strategic, organizational, and business aspects.

The goal of the *enterprise*[1] *framework* is to characterize the global environment in which system evolution occurs and provide insight into a wide range of management and technical issues that must be addressed in evolving software-intensive systems. The objective is to help managers identify critical success factors and develop a synergistic set of management and technical practices for planning, evaluating, and managing system evolution initiatives.[2] These system evolution efforts may range from a series of incremental improvements (e.g., adding new security features, incorporating a new database management system, modifying the software to properly process dates for the year 2000 and beyond) to reengineering the entire system. Reengineering can be viewed as one form of system evolution where more drastic or extensive measures (e.g., a major system reconstruction or rehosting effort) may be required to further evolve the system due to its brittleness, complexity, or general state of ill health.

While the framework is aimed primarily at managers at the organizational unit and project level, its utility extends across the enterprise by virtue of its global perspective. This global or enterprise perspective is intended to reflect a holistic approach to system evolution that aligns the organizational, engineering, technology, and system elements with the strategic and business operations of the enterprise.

[1] An enterprise may be a company or government agency, or an organization within a company or government agency, or it may span several organizations, companies, or government agencies that have common business ties and interests.

[2] The phrase "system evolution initiative" is used to connote a formally established and organized effort to evolve a system, and is used interchangeably with the phrase "system evolution effort."

By calling this framework an enterprise framework, we are not suggesting that the entire enterprise must be involved in every aspect of a system evolution initiative, or that senior management must be involved every time a system evolution task or activity is performed or an issue arises. However, high-level managers must be involved in certain aspects of the planning and concept formalization phase. In fact, these managers may intervene in the system evolution activities when an issue arises that is beyond the scope, authority, or ability of the project or organization[3] to resolve.

The framework portrays the enterprise-wide context in which a system evolution initiative occurs (and in which the responsible organization and project function). The framework's structure reflects a global characterization of the system evolution problem space and solution space. Included are a set of checklists that cover a wide range of issues. The checklists are designed to probe the strategic direction and values of the enterprise, the broad community and customer needs the enterprise serves, and the organizational units and teams that execute its missions. In summary, what makes the framework an "enterprise framework" is not the degree to which the entire enterprise is involved, but its perspective.

An enterprise approach promotes a unified effort by a parent organization and its project to address the impact of system evolution on other affected organizations, the business environment, enterprise-wide customers and users, and interfacing systems.

[3] The term "organization" is used to indicate the corporate unit that is responsible for managing multiple system evolution projects.

2. Framework Structure

The enterprise framework consists of seven elements that are building blocks for a successful system evolution or reengineering effort. Each element has a critical set of technical and management issues that are essential for developing a comprehensive plan of action.

The elements of the framework are

- organization
- project
- legacy system
- systems engineering
- software engineering
- technologies
- target system

Figure 1 is a high-level graphical representation of the enterprise framework that identifies the elements that managers and practitioners need to consider in a system evolution initiative. The arrows are indicative of how each of these elements uniquely contributes to a system evolution initiative.

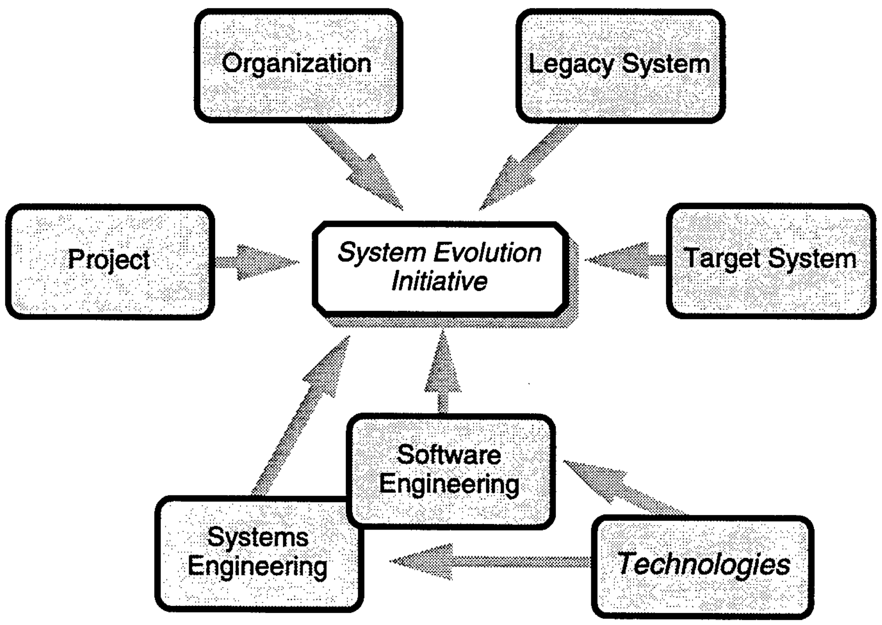


Figure 1: A Framework for the Disciplined Evolution of Legacy Systems

The structure and composition of these framework elements are described in more detail in Sections 3 through 6. For each element, one or more checklists are provided to guide the analysis of that element during an actual system evolution initiative and provide insight into critical success factors.

2.1 Relationship of Framework Elements

The elements of the framework are applicable to a wide class of system evolution initiatives. In the workplace, however, the specific composition of the framework elements and their interrelationships are a function of the particular enterprise, its culture, and its management and technical practices.[4]

One of the ways the framework promotes an understanding of these elements and their interrelationships is through considering the framework's representative set of practices and examining the issues that are raised in the framework's checklists. These practices and issues are identified in the sections that describe each of the framework's elements.

Many example practices and issues are identified to draw out what the roles and specific modus operandi of the organization and project will be in evolving the system. Use of the checklists provides insight into who the stakeholders and decision makers are, and what enterprise factors (and work products) govern the tasks and the decision-making processes. The checklists may also surface "gray areas" in the enterprise planning such as how interdependent aspects of the work will be coordinated with external organizations and customers, and how the proposed system will potentially be affected by (or affect) other enterprise efforts that are already underway or in the planning stages.

[4] We use the term "practices" to refer to the life-cycle activities, processes, and work products that are used to carry out the system evolution tasks described in the project plan and migration strategy.

One example of an enterprise view of a system evolution initiative (based on the enterprise framework) is shown in Figure 2. This "big picture" view of a system evolution effort illustrates the role of the framework elements and their interrelationships and how they can be integrated to form a system evolution cycle. This cycle begins with the customer negotiating with the organization and ends with the improved products and services being provided to the end user. In essence, this diagram represents the high-level enterprise architecture for performing system evolution.

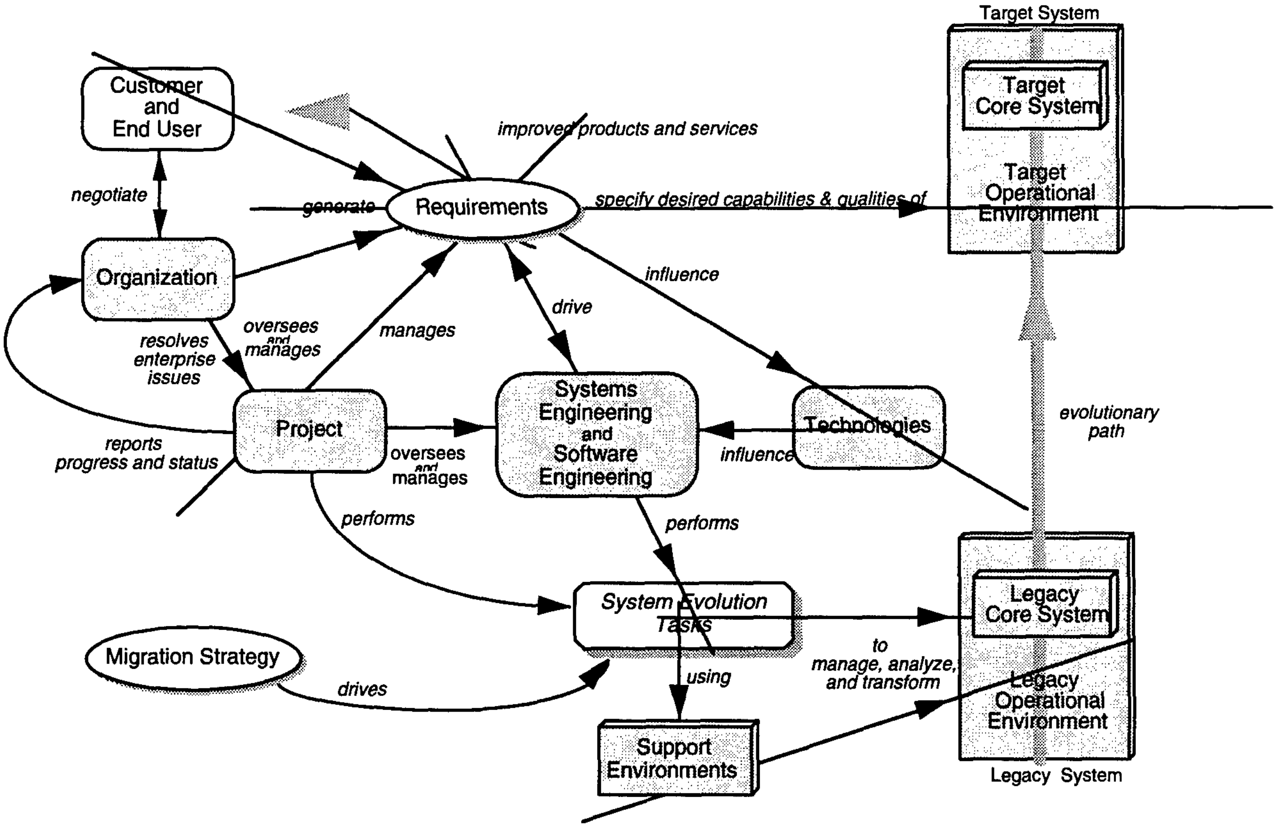


Figure 2: An Enterprise View of a System Evolution Initiative

In Figure 2, the shaded objects correspond to the seven elements of the enterprise framework. The primary role of each element is noted on the arrows that depict the interrelationships of the framework elements.

The focal point of this enterprise view of a system evolution initiative is the non-shaded object labeled "System Evolution Tasks." Conceptually, the transformations required to evolve the existing system to the desired system state are specified in terms of a set of tasks representing good systems engineering and software engineering practice. In practice, these tasks are driven by the selected migration strategy that is traditionally described in the project plan. The tasks may be performed by the project team, the systems and software engineering teams, other organizational units of the enterprise, or their contractors.

Examples of system evolution initiatives may include incremental enhancements to a system, rehosting the software applications on a new platform, or reengineering a system to incorporate new system capabilities, such as added functionality, security features, or fault-tolerant processing. In the rehosting example, the system evolution tasks may include analyzing code to determine system dependencies, wrapping code, developing I/O drivers, and performing regression tests. In the reengineering example, system evolution tasks may include developing a domain model, developing a concept of operations, prototyping, developing a new system architecture, and enhancing existing application programs.

The other non-shaded objects in this enterprise view represent important nodes in the interrelationships of the framework elements. Two of these nodes, "Requirements" and "Migration Strategy," are major work products that play a significant role in planning and also drive the system design and implementation. The other node represents the "Customer and End User." The interests of the customer and user are addressed explicitly in several of the checklists for the individual framework elements.

Giving careful consideration to the roles and contributions of the framework elements and understanding their interrelationships in a particular organizational setting is an essential step in achieving an integrated and coordinated approach to system evolution.

3. Legacy System

System evolution begins with the legacy system which, as shown in Figure 3, includes the legacy core system, its operational environment, and its support environments. Each of these are described in the following sections.

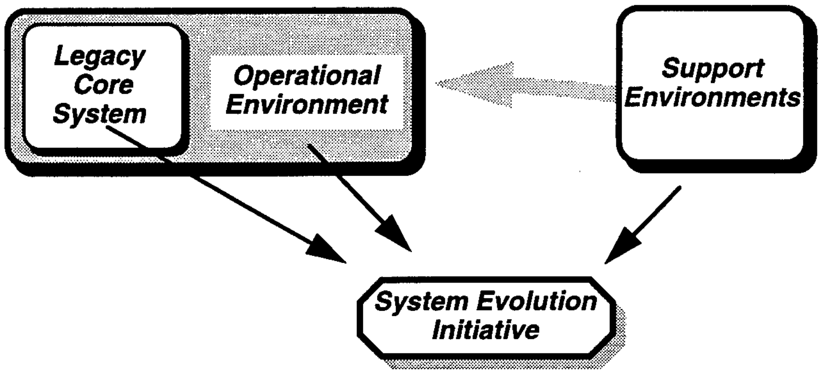


Figure 3: The Legacy System Element

3.1 Legacy Core System

The legacy core system is an operational "software-intensive" system that is a candidate for evolutionary improvement. As shown in Figure 4, the system can be characterized in terms of technical factors, such as its architecture, products and services, functionality, usability, and other quality attributes.

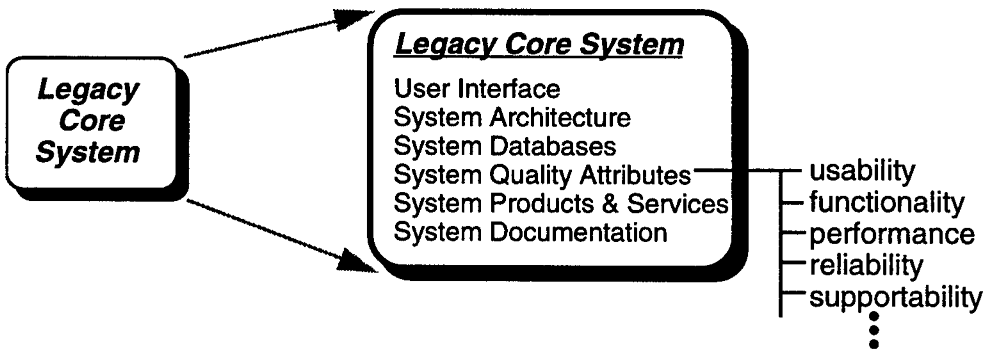


Figure 4: Legacy Core System

The challenge in the disciplined evolution of systems is understanding the functionality, design, operation, and performance of the legacy system and anticipating the types of changes that will be required over the useful life of the system. After years of maintaining, upgrading, and enhancing the legacy system, the user manuals and system design documentation are often out of date, inaccurate, and fail to reflect the current system's capabilities and operation.

The following questions form an initial checklist for probing the technical features and current state of the legacy system:

- Is there a current system configuration diagram? A system design document?
- Are the software architecture and software design well documented?
- Are the system interfaces and communication protocols documented?
- What are the dependencies on external interfaces?
- Is the functionality and operation of the system described adequately in user and system documentation?
- Have all the user interfaces been identified?
- Have the software applications and critical algorithms been identified? Have they been analyzed?
- Are the software interfaces and message and data formats documented?
- Have the performance characteristics of the system been assessed? Have benchmarks been run?
- Are the source code, library elements, and build scripts available? Are they current?
- Is there documentation on the logical and physical data dictionaries?
- Have dependencies on undocumented features been identified?
- Have the complexity and brittleness of the system been assessed?
- Has the integrity of the system been affected adversely by the maintenance legacy?
- How stable is the system's operation? Have the unresolved problem reports and change requests been reviewed for trend information?

3.2 Legacy System Operational Environment

The operational environment includes the network of computer resources, the customer and user sites being supported, and the interfacing systems, as indicated in Figure 5.

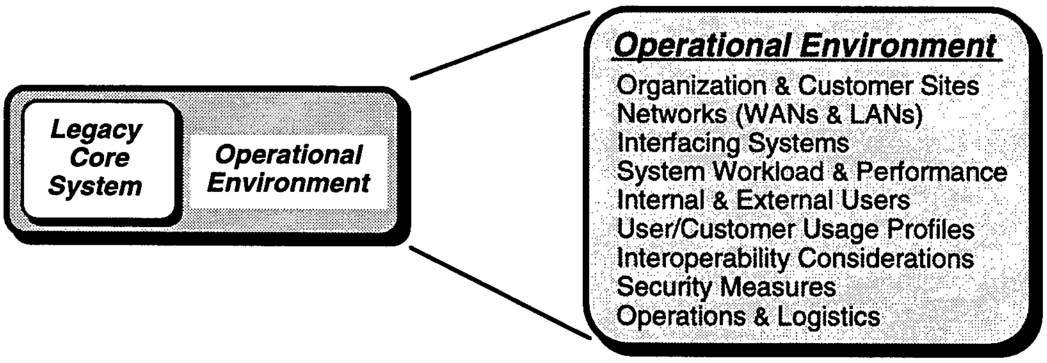


Figure 5: Legacy System Operational Environment

The operational environment can be characterized by the number of interfacing systems, number of users being supported, the system workload, the number and type of applications being used, user dependencies on artifacts produced by the system, system operations, and hardware and software interdependencies.

A checklist for defining a baseline of the legacy system's operational environment can include the following:

- Are all of the customers, customer sites, and user groups identified?
- Are all of the legacy system products and services on which the users depend identified?
- Is there a profile to accurately characterize the current system workload?
- Are all of the external artifacts, system files, and procedures on which the users depend identified?
- Are there operational usage scenarios to ensure that there is a common understanding of the system's capabilities and operation from a user's viewpoint?
- Is there an accurate, up-to-date network configuration diagram that specifies the subsystems and their interfaces?
- Are all of the external system interfaces identifiable and documented?
- Are the hardware and software interoperability dependencies with external sites identified and documented?
- Are the software communication protocols identified? Are they documented?

- Are the system's security provisions and features clearly understood by the project team?
- Are the logistic, support, and system administration operations (and roles and responsibilities) itemized? Are they traceable to specific subsystems (and agents)?
- Will the operation of the legacy system be sustained to allow adequate time for users to obtain training and fully make the transition to the proposed system?

3.3 Legacy System Support Environments

Multiple support environments (shown in Figure 6) are frequently employed in managing, developing, maintaining, and sustaining a system over its life cycle. These support environments may include a mix of environments for development and maintenance, test and integration, project management, and other support functions.

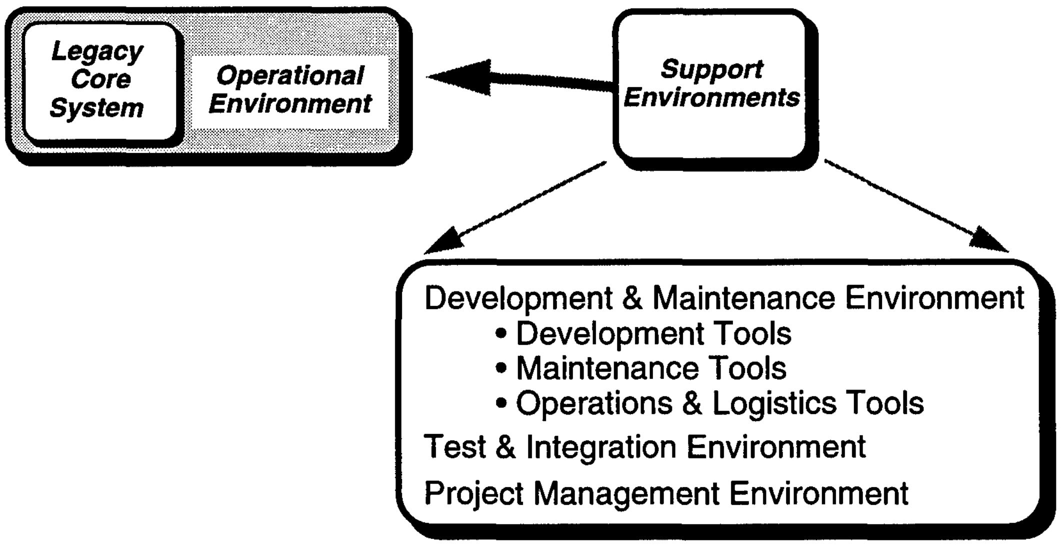

Legacy Core System
Operational Environment
Support Environments
Development & Maintenance Environment
• Development Tools
• Maintenance Tools
• Operations & Logistics Tools
Test & Integration Environment
Project Management Environment

Figure 6: Legacy System Support Environments

Like the legacy system, these environments evolve over time as the system matures and new needs arise. For example, the focus of the development and maintenance environment is initially on development tools, but after the legacy system becomes operational, the focus shifts toward tools for analyzing and maintaining the system, performing selective system upgrades and testing, and sustaining the day-to-day operation of the system. However, maintaining the core capabilities of the original development tools is critical to being able to enhance and evolve the system in an effective and disciplined manner.

An important consideration in establishing support environments is whether there will be a separate environment for integration, test, and troubleshooting or whether the project will have to contend with scheduling the actual operational (i.e., legacy) system to perform these essential functions. Still another aspect of support environments is the degree to which automated tool support is provided for routine project management and support functions such as configuration management and quality assurance.

The checklist for the support environments should include the following.

- What is the composition of the support environments? What products and services do they provide?
- To what extent is the development and maintenance environment consistent with the developer's original environment? Does it include the tools used for requirements elicitation and validation, design, and testing?
- To what extent are the tools in the support environments integrated? Are there established procedures for their use?
- Do the tools enforce or promote good programming practices? Are there documented programming guidelines and practices? Are metrics automatically collected or retrievable?
- Is a separate integration and test environment available to the maintainers apart from the operational system? Does this environment accurately reflect the operation of the legacy system?
- Are project management functions, such as planning, estimating, costing, scheduling, progress reporting, issue and problem resolution, supported? To what degree are they supported?
- Which functions are supported by automated tools? Are some labor-intensive functions being performed manually? Can they be improved by adopting new tools or processes?
- How is configuration management being performed on the hardware and software products undergoing development, reengineering, or maintenance? How is quality assurance being performed? Are the efforts coordinated?
- Are software build processes well documented? Do they produce repeatable results?
- Does the integration and test environment provide an automated regression testing capability?
- How are new releases placed into operation?
- To what extent are proprietary or customized tools being used?
- Have commercial off-the-shelf (COTS) tools been updated? Are the versions of the tools still supported by the tool vendor? Are the licenses up to date?
- Is there a defined process for determining when COTS tools should be upgraded to the vendor's latest product release?
- Are the support environments themselves under configuration management and control?

Resolving these issues corrects shortcomings of the support environments that could derail a system evolution effort.

4. Organization and Project

The organization and project are the next elements of the enterprise framework. They play a key role because the real barriers to success are frequently not technical, but are related to management and culture. These barriers are a function of the particular organization, its project structure, and its practices. The structure of both of these framework elements corresponds to the activities they perform, the infrastructure support they provide, the processes they use, and the work products they produce (see Figures 7 and 8).

4.1 Organization

We use the term "organization" to denote a structural unit within a company, or government agency, that is responsible for managing a group of projects in support of the business operations and mission of the enterprise. In this context, the organization aligns projects to meet customer needs and the strategic goals and objectives of the enterprise. The organization is responsible for chartering projects, establishing project goals and objectives, and empowering the projects to carry out the mission defined for them.

Organizational management activities include overseeing and guiding the system evolution and reengineering project efforts, providing suitable infrastructure support, promoting enterprise-wide coordination, and resolving high-level issues and conflicts. Figure 7 is an overview of a representative set of management activities, infrastructure support, processes, and work products for the organization element of the framework.

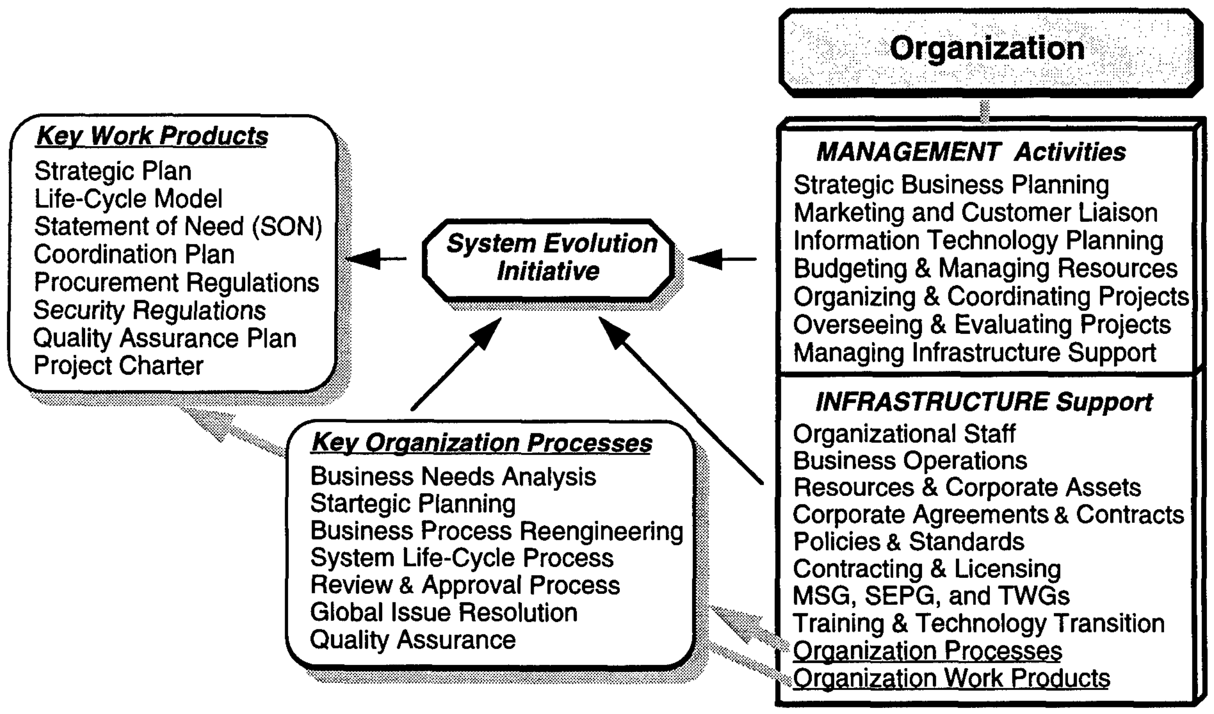

Organization
MANAGEMENT Activities
Strategic Business Planning
Marketing and Customer Liaison
Information Technology Planning
Budgeting & Managing Resources
Organizing & Coordinating Projects
Overseeing & Evaluating Projects
Managing Infrastructure Support
INFRASTRUCTURE Support
Organizational Staff
Business Operations
Resources & Corporate Assets
Corporate Agreements & Contracts
Policies & Standards
Contracting & Licensing
MSG, SEPG, and TWGs
Training & Technology Transition
Organization Processes
Organization Work Products
System Evolution Initiative
Key Work Products
Strategic Plan
Life-Cycle Model
Statement of Need (SON)
Coordination Plan
Procurement Regulations
Security Regulations
Quality Assurance Plan
Project Charter
Key Organization Processes
Business Needs Analysis
Startegic Planning
Business Process Reengineering
System Life-Cycle Process
Review & Approval Process
Global Issue Resolution
Quality Assurance

Figure 7: Overview of the Enterprise Framework's Organization Element

Some of the key organization processes are business and mission needs analysis, business process reengineering, and a life-cycle process. Example work products the organization develops include a strategic plan, technology plan, statement of need, and coordination plan.

An initial set of questions that help probe the organization element includes the following:

- What are the enterprise goals?
- Has a common vision been developed and communicated?
- Have the key decision makers and stakeholders been identified?
- Are the goals of the organization aligned with the enterprise goals?
- Are there defined criteria for the successful accomplishment of goals? Are these criteria measurable?
- What is the corporate information technology strategy?
- What is the overall scope of the system evolution effort?

- Is there an established procedure for performing business/mission needs analysis to determine how new customer needs can best be met?
- Are the roles and responsibilities of each of the organizational units involved in the system evolution effort well defined?
- How will efforts be coordinated across organizational units and with external customers?
- Does the organization provide suitable infrastructure support to assist projects in contracting, quality assurance, and other key activities that may be beyond the scope of an individual project to perform?
- What is the review and approval process for new and revised work products?
- Is there a well-defined issue-resolution process?

In addition, there are things that organizations commonly tend to do, but should avoid doing. A representative checklist for intercepting bad practices includes the following questions:

- Have the benefits of evolving the legacy system been predetermined without first conducting a thorough analysis?
- Has the feasibility of evolving the system also been predetermined?
- Have all three project variables (capability, schedule, and cost) been determined by the organization prior to having the project develop a formal plan for evolving the system?
- Have some aspects of the solution space been predetermined before analyzing the system and involving the project team?
- Has sufficient time been allowed for a thorough systems engineering analysis before finalizing the project implementation plan?
- Is a complete project implementation plan being required before developing a concept of operations for the target system and obtaining the agreement of customers and the user community?
- Are a new and unproved life-cycle process being mandated without *soliciting* project feedback and the agreement of project team leaders?
- Is training and other infrastructure support being provided for piloting the application of new processes, tools, and work products, before attempting to institutionalize them?

While these practices fall under the category "to be avoided at all cost" and may be recognized universally as having severe consequences, organizational experience suggests that they are commonplace and may be endemic to certain environments. Avoiding these practices is especially relevant to taking an enterprise approach to system evolution, because these practices can not be mitigated by the project and do not lend themselves to a technical solution. The decision for partitioning and assigning decision rights is the prerogative of the organization [Jensen 83].

4.2 Project

The term "project" denotes a structural unit within an organization that is responsible for evolving a system that provides products and services to the organization and its customers. In this context, the project is responsible for planning and structuring the system evolution effort, organizing the tasks and people, overseeing the systems engineering and software engineering activities, and managing the work to ensure that the products and services they produce meet the users' needs and fulfill the organization's goals and objectives. The project element of the framework consists of management activities, infrastructure support, key work products, and key processes, as illustrated in Figure 8.

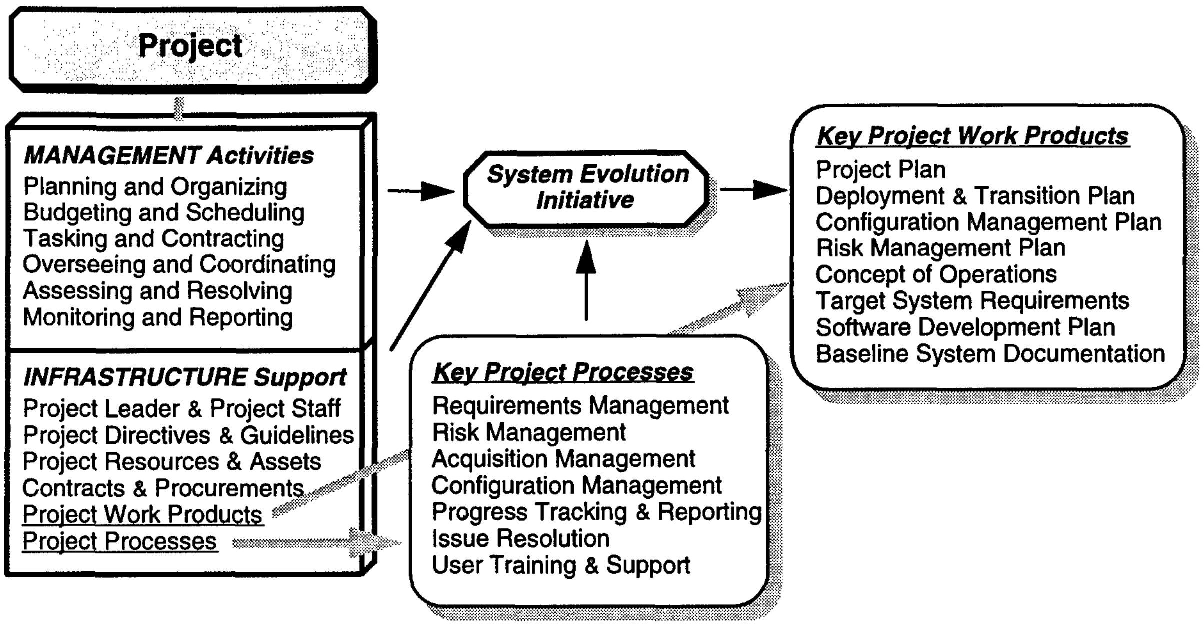


Figure 8: Overview of the Enterprise Framework's Project Element

Project management includes activities such as planning and estimating, tasking, financial tracking, contracting, risk management, project tracking, progress reporting, configuration management, and issue resolution.

One example of a key project work product is a project plan. A project typically follows an approved life-cycle model (an organizational work product) to develop a work breakdown structure (WBS) that defines the system evolution or reengineering tasks. These WBS tasks, which are an integral part of the project plan, cover all aspects of the project (including project management activities, training, contracting, development, reengineering, integration and testing, installation, and transition) to operational use. In the process of executing the plan and conducting project reviews, many issues will surface that must be resolved. As shown in Figure 8, a progress tracking and reporting process and an issue-resolution process are two examples of key project processes.

The project infrastructure consists of a set of resources, assets (e.g., facilities, tools, processes, work products), and supporting services that enable a project to perform its mission efficiently. In addition to prescribed processes and work products, infrastructure support may include project policies, directives and guidelines, and contracts for obtaining additional resources or procuring specific products and services.

A checklist for probing the project element includes the following questions.

Planning Related

- Is there a clear understanding of the organization's goals and a linkage between the organization's strategy and the project's strategy?
- Is there a comprehensive project plan? Are all the deliverables specified?
- Is ownership of each plan and project work product established clearly?
- Are roles and responsibilities defined clearly?
- Does the project plan define the migration strategy clearly? Are the systems and software engineering teams fully supportive of the migration strategy?
- How realistic is the project plan and work breakdown structure (WBS)?
- Does the WBS describe all the tasks for implementing the migration strategy?
- Does the plan include estimates of the resources and time required for each task?
- Are there subsidiary plans covering risk management, configuration management, quality assurance, and software development? Have the plans been suitably coordinated?
- What are the cost and schedule for completing the effort?
- Is a network activity diagram included which identifies the intertask dependencies?
- How will the project obtain and integrate the necessary interdisciplinary skills?
- What kinds of infrastructure support do the systems and software engineering activities require from the project? Are they included in the project plan?
- Has training been arranged for the system developers and software engineers?
- Are all phases of the project's life cycle addressed adequately in the project plan?
- How will progress be measured and reported?
- Is there a process in place to ensure that the project plan is updated as changes occur?
- Is there a chief systems engineer, or group, who is accountable for the systems engineering and software engineering effort?
- Will a project team composed of key task leaders and interdisciplinary engineers be established to serve as a system design team? If not, how will global systems engineering issues and specialty engineering requirements (e.g., security) be addressed and coordinated adequately?
- Do plans include training for customers and users of the system?

Risk Related

- How will risks be managed and mitigated?
- Are a process and criteria in place for make/buy decisions?
- Has an effective contracting strategy been developed?
- Is the project adequately funded?
- Is there evidence of overly optimistic schedule compression?

Requirements Related

- Has a common concept of operations for the proposed system been developed and communicated?
- Does the project have a requirements change management process?
- How are the customer and user requirements prioritized?

5. Systems Engineering and Software Engineering

The next elements of the enterprise framework are systems engineering and software engineering. As shown in Figure 9, both elements are characterized in terms of a fundamental set of activities that are indicative of the scope of these two core disciplines. Underlying these activities is another set of processes and work products which is not elaborated in this report. Insight into the nature of these supporting processes and work products can be obtained from a review of the Systems Engineering Capability Maturity Model^{SM} (SE-CMM®) [SECMM 95] and Capability Maturity Model for Software (SW-CMM) [Paulk 93], both developed by the SEI.[5]

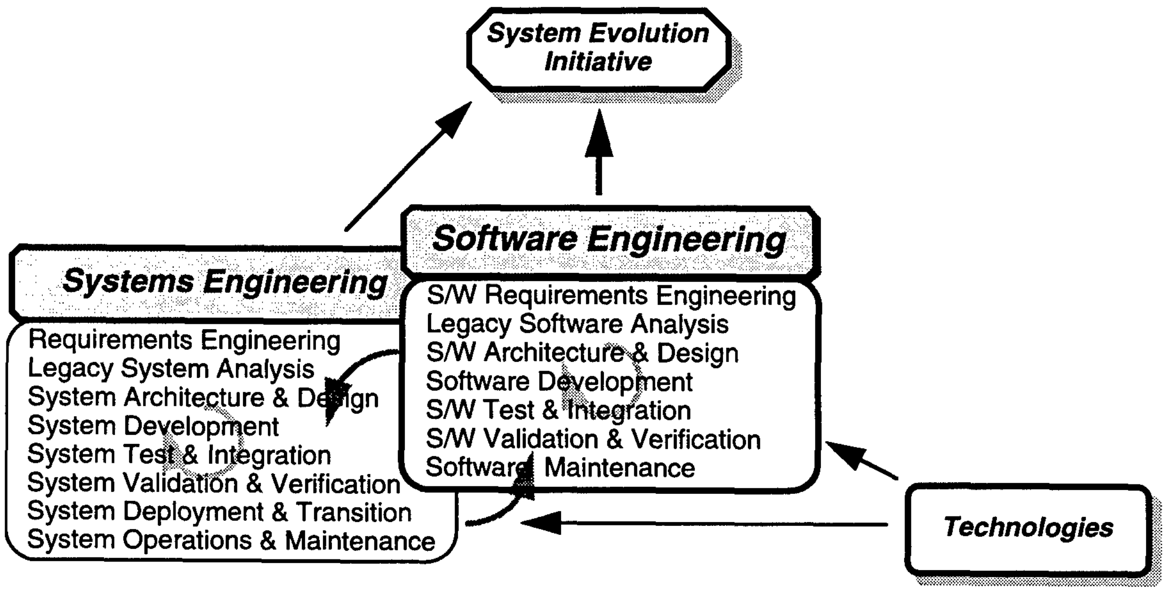

System Evolution Initiative
Systems Engineering
Requirements Engineering
Legacy System Analysis
System Architecture & Design
System Development
System Test & Integration
System Validation & Verification
System Deployment & Transition
System Operations & Maintenance
Software Engineering
S/W Requirements Engineering
Legacy Software Analysis
S/W Architecture & Design
Software Development
S/W Test & Integration
S/W Validation & Verification
Software Maintenance
Technologies

Figure 9: Systems and Software Engineering Elements

The high-level systems engineering activities itemized in Figure 9 correspond closely to the seven process areas (PAs) of the "Engineering Category" of the SE-CMM. We added two activities: "Legacy System Analysis" and "Systems Operation, Maintenance, and Sustainment," which reflect a system evolution focus.

[5] ^{SM}Capability Maturity Model is a service mark of Carnegie Mellon University.

® Registered in the U.S. Patent and Trademark Office.

The high-level software engineering activities itemized in Figure 9 parallel the system engineering activities to emphasize their interrelationship and the importance of an integrated and synergistic approach to systems and software engineering. Likewise, including the "Legacy Software Analysis" activity reflects a major thrust of legacy software analysis (i.e., assessing the evolvability of a legacy system) [Brown 96]. The software engineering activities do not correspond directly with the key process areas of the SW-CMM because of the difference in emphasis (i.e., system evolution characterization vs. software development capability assessment and process improvement). However, the key process areas (KPAs) and key practices described in the SW-CMM provide invaluable insight into generic software practices that are essential to good software engineering and system evolution.

A useful checklist in carrying out the systems engineering and software engineering activities (in conjunction with the target system element checklists) includes the following questions.

- Are mechanisms in place to ensure that software engineering tradeoffs and considerations are an integral part of the up-front systems engineering activities?
- Has an incremental development strategy been adopted?
- Has consideration been given to adopting an incremental implementation approach that is driven by the highest priority risks that have been identified to date?
- To what extent will prototyping be employed? Have criteria been established?
- Is there a defined process for performing system engineering tradeoff analyses and allocating system requirements to hardware and software?
- Is there a formal process for risk assessment and mitigation? Is it performed regularly or is it a one-time activity?
- Are appropriate systems and software engineering tools being used?
- What means are being employed to ensure requirements traceability?
- Is the systems engineering team responsible for the technical oversight of individual hardware and software product developments? How will this oversight be accomplished?
- How will the degree to which the legacy system software is salvageable and evolvable (from a technical and economical standpoint) be determined?
- Is there evidence to support that the prescribed systems and software engineering methodologies are effective?
- Is a process in place for evaluating candidate software architectures and assessing their quality attributes?
- What approach is planned to acquire an understanding of the design, functionality, usability, reliability, performance, and operation of the legacy system?
- What is the process for deciding to make changes to programming languages, operating systems, and related technologies?
- Are programming guidelines established? Are they followed?

- Is there a change-management strategy for accommodating ongoing software changes to the legacy system that occur during the development of the target system?
- Are the transition issues associated with operationally deploying the system being addressed?
- Is there a strategy in place for achieving upward software compatibility?
- How will changes to software interfaces with external systems be coordinated?
- Are programs needed for converting existing data files and databases? Will they automatically make the conversion or will user intervention be required?
- Are the training needs of the systems engineers and software engineers identified?

6. Technologies

The next element of the framework covers the technologies being considered for the proposed (target) system. Evolutionary changes are frequently driven by promising new technologies that will accommodate the following:

- meeting new mission and business processing needs
- overcoming technical obsolescence
- countering increased maintenance costs

New technologies may have a significant impact on how software and systems engineering is performed. One example of a technology with the potential to profoundly change the ways in which software systems evolve over time is wrapping (i.e., encapsulation) and other distributed object technologies [Weiderman 97].

Example technologies that may be relevant to the proposed system include the following:

- object-oriented design
- product line architectures
- Common Object Request Broker Architecture (CORBA)
- fault-tolerant computing
- Rule-Based Intrusion Detection (RBID)
- Java programming language

New technologies may also apply to the legacy system support environments to aid engineers in analyzing and testing the legacy system. This testing helps engineers to better understand the system's capabilities (and quality features) and assess the impact of the proposed changes. Example technologies that may be relevant include the following:

- domain engineering
- Software Architecture Analysis Method (SAAM)
- code analyzers and visualization tools
- performance and impact analysis tools
- object-oriented analysis
- transformation tools

Insight into other candidate technologies can be obtained from a recently published Software Technology Reference Guide [CMU/SEI-97-HB-001]. This guide identifies a spectrum of technologies that could be relevant to system evolution. The guide addresses technology in its broadest sense and includes information on approximately 60 software technologies.

A checklist for screening new technologies for potential project application might include the following issues:

- Does the technology have the potential to make a significant contribution to the enterprise goals and objectives? Can it provide a competitive advantage?
- Is the technology a prerequisite for the system evolution effort?
- Is the technology sufficiently mature and stable?
- What tangible benefits can the technology provide? Is it required for system compatibility? Is it a prerequisite for adopting other technologies?
- Have pilot efforts or case studies confirmed the suitability of the technology for the specific application domain?
- Have the benefits of adopting the candidate technology been quantified?
- What is the potential impact of not adopting the technology?

Once a particular technology has been determined to be generally suitable, a checklist covering the technology selection process should include answers to the following questions:

- Is the cost, schedule, and impact of applying the new technology acceptable?
- Is adequate training available? Are key members of the project team already well versed in the technology? Can they act as mentors to other team members?
- Have the pros and cons of alternative technologies been weighed carefully (preferably using a formal risk assessment process)?
- Has the impact of the new technology on existing customers and users been analyzed? Do the customers and users have any strenuous objections? Or unheeded cautions?
- Is management aware of the technology adoption plans? Are these plans consistent with the organization's strategic plan? Are there any reservations or cautions?
- Is a suitable measurement program being adopted to quantify and evaluate the actual benefit of applying the technology?
- Is there a contingency plan in the event that any unforeseen technology "show-stoppers" arise?

7. Target System

The final element of the enterprise framework is the target system (shown in Figure 10). The target system consists of the target core system, the target operational environment, and the target support environments. Since the legacy and target systems represent a "before" and "after" picture of the reengineered system, the elements of the target system closely mirror those of the legacy system.

The Concept of Operations shown in Figure 10 is a high-level requirements document that describes the capabilities and operation of the proposed target system in user terms. This document enables the up-front "buy-in" of customers and users. Organizational management can also use a Concept of Operations to verify that the proposed system is consistent with the goals and objectives of the enterprise.

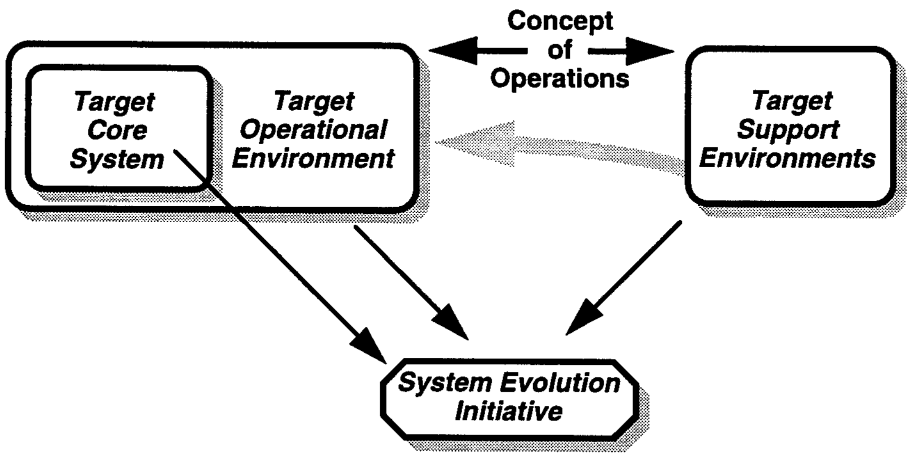

Concept of Operations
Target Core System
Target Operational Environment
Target Support Environments
System Evolution Initiative

Figure 10: Target System Element

7.1 Target Core System

The target system reflects the desired system state. Some of the salient characteristics of the target core system, operational environment, and support environments are identified in Figure 11.

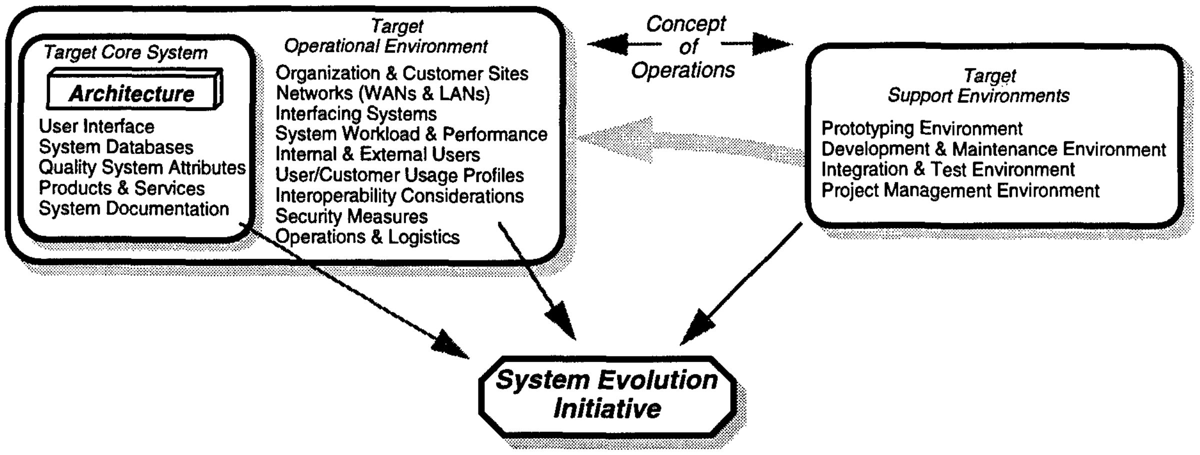

Target Core System
Architecture
User Interface
System Databases
Quality System Attributes
Products & Services
System Documentation
Target Operational Environment
Organization & Customer Sites
Networks (WANs & LANs)
Interfacing Systems
System Workload & Performance
Internal & External Users
User/Customer Usage Profiles
Interoperability Considerations
Security Measures
Operations & Logistics
Concept of Operations
Target Support Environments
Prototyping Environment
Development & Maintenance Environment
Integration & Test Environment
Project Management Environment
System Evolution Initiative

Figure 11: Characteristics of Target System Element

Decisions about the target system involve tradeoffs that represent a compromise between the desired state, what is known about the legacy system, and the available resources. A checklist of issues to consider when making these decisions includes the following:

- Is there a prescribed means for eliciting and validating the target system requirements? Has it been used before? Is there evidence of its effectiveness?
- Is there a Concept of Operations to describe the proposed target system?
- Have operational scenarios been developed to describe how the proposed system will operate?
- Have the Concept of Operations and operational scenarios been validated with customers, users, and key systems personnel?
- Is the difference between the current "virtual requirements" of the legacy system and the new target system requirements well understood?
- Are there standards with which the target system must comply?
- What ground rules have been established for the use of COTS software?
- How robust is the current legacy system architecture? Is it practical to evolve this architecture to meet the target system requirements?
- Should the system be rehosted on a new platform or operating system? Is the use of a new programming language justified?

- What process is used to determine the target system architecture *requirements*?
- What are the desired performance, availability, and security *attributes*?
- Can the target system be evolved incrementally over a period of time? Or is a major reengineering effort required to bring about the desired changes?

7.2 Target Operational Environment

The target operational environment includes (1) the global and local networks the target system will be part of and (2) the interfacing systems and subsystems at customer and organizational sites. Characteristics of the operational environment include the number and types of users (internal and external), their usage of the system's products and services, the projected system workload and performance, and any formal agreements or contractual commitments that may apply.

If the target system uses totally new computing resources or requires significant changes on the part of the user, the target system may not replace the legacy system quickly. Deploying the reengineered system may then constitute a formidable systems integration problem.

A checklist of issues to consider includes the following:

- What changes are required in the operational environment to accommodate the new target system requirements?
- What is the projected impact of the proposed changes on current business operations? How will these affect the customer and the organization?
- Do the customer and user requirements include explicit changes to the operational environment? How do these changes affect the target system (hardware and software)?
- What is the projected impact of the proposed changes on performance and availability?
- What differences are there between the existing legacy system environment and the proposed target environment? Are there incompatibilities that will need to be resolved?
- Will support for some of the existing products and services be dropped? What customers and users will be affected?
- Which external interfaces need to be modified? How will these modifications be coordinated with external systems and users?
- What testing is needed to assure interoperability?
- What is the plan for "roll out" and "cut-over" to the new system?
- What parts of the target and legacy systems need to coexist during operational transition?
- In the event of a crisis, to what degree can support be rolled back to the legacy operational environment?
- Will the new target environment impose new operating procedures?
- Will operators or system administrators require training on the new operating environment?
- Have training needs been identified for customers and users of the system?

7.3 Target Support Environments

The target support environments provide the project team with the facilities and tools (and associated methods and procedures) to carry out their respective project management, systems engineering, and software engineering responsibilities. This support includes performing such technical activities as prototyping, developing, integrating, testing, and documenting the target system and its software. Useful in supporting project management would be tools for generating activity network diagrams for WBS tasks, estimating software size and cost, creating project schedules, collecting progress measures, and generating reports. Depending on the scope and magnitude of the desired system changes, the target support environment may be just an evolutionary outgrowth of the legacy support environment, or it may be radically different and require a totally new development (and procurement) effort. Be aware of the degree of variance and understand the ramifications.

The checklist in Section 3.3 on legacy support environments also pertains to target support environments. In addition, issues of compatibility, upgrading, and integration between the legacy and target environments must be addressed.

8. Using the Enterprise Framework to Unify System Evolution

Figure 12 is a composite all of the framework elements and depicts the intricacies and complexities a manager may contend with in evolving software-intensive systems. This evolution extends from concept development through deployment and affects everyday operation and maintenance. This expanded view includes a characterization of the legacy and target system elements and a representative set of activities, key processes, and work products that characterize an enterprise-wide approach. Figure 12 illustrates the need for a disciplined approach to system evolution to ensure that the many diverse activities, processes, and work products are suitably coordinated and integrated into a cohesive plan of action.

While the actual set of activities, processes, work products, and other relevant factors may vary substantially from organization to organization and project to project, the *crucial enterprise-level questions* for any system evolution project are the following:

- Is an enterprise-wide business and technical strategy in place? Has it been communicated to all affected parties?
- Are roles and responsibilities clearly defined?
- Are the enterprise-wide activities, processes, and work products identifiable? Are they adequately described and understood? Do they reflect a unified approach consistent with the organization's life-cycle model?
- If one or more of the framework activities is not covered, what is the default condition? What impact will using the default condition have?
- Are a set of activities, processes, and work products comparable to those shown in Figure 12 being specified? Are they adequate?
- If some processes or work products are not being addressed, is it an oversight? Is it indicative of a problem? What are the risks?
- Has ownership and accountability for the activities, processes, and work products been clearly established?

In the absence of an “enterprise” type of approach to system evolution:

- How will global issues be resolved?
- How will priorities be determined?
- How will enterprise-wide coordination be ensured?
- How will work products from other ongoing (or planned) efforts be leveraged?
- How will progress and quality be assessed?
- How will lessons learned be captured and communicated?
- How will practices be improved?

By drawing out these important enterprise issues early in the planning cycle, an organization can use the framework and its checklists to guide the development of an integrated set of management and technical practices for a spectrum of system evolution activities.

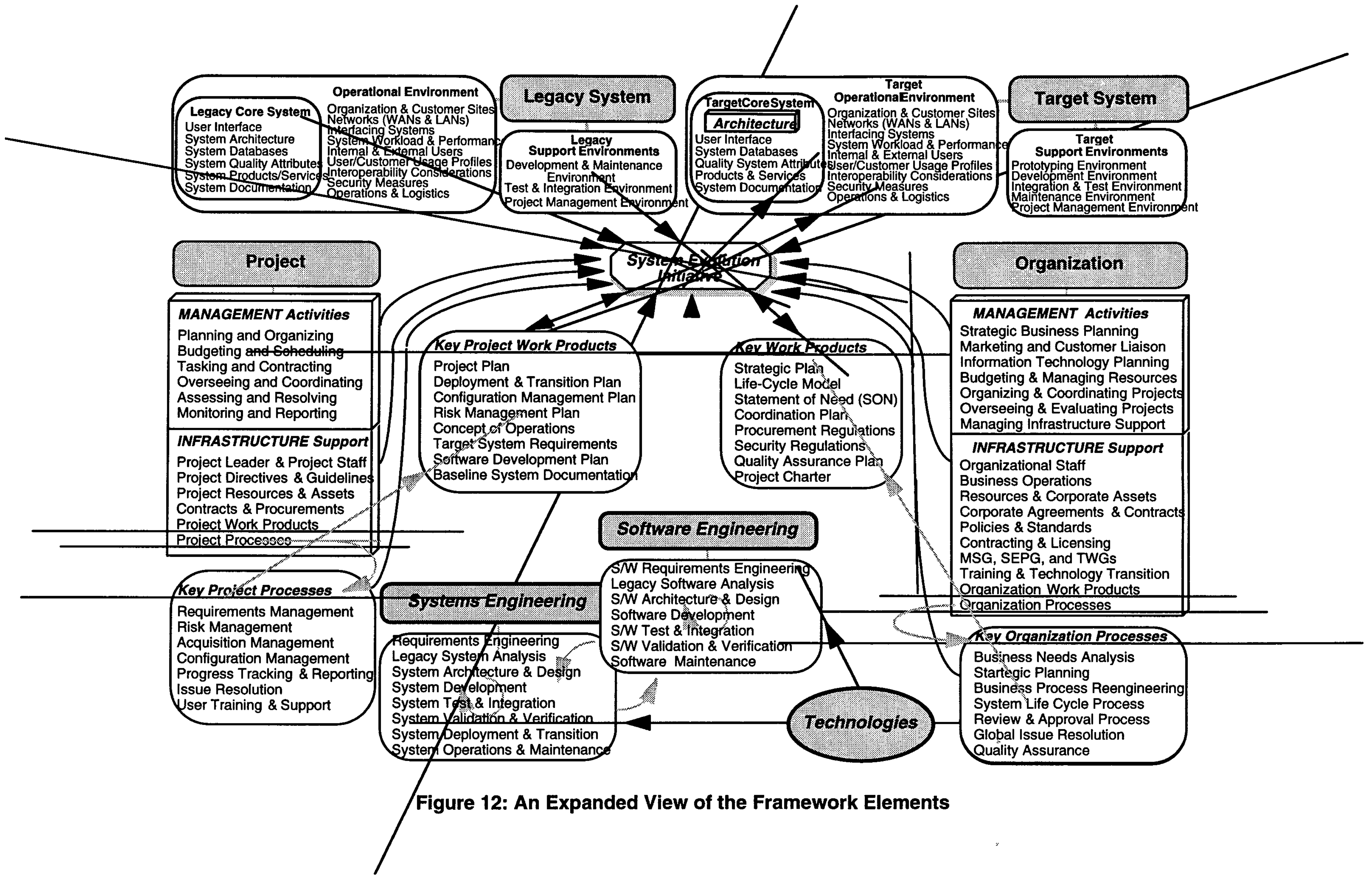

Operational Environment
Legacy Core System
User Interface
System Architecture
System Databases
System Quality Attributes
System Products/Services
System Documentation
Organization & Customer Sites
Networks (WANs & LANs)
Interfacing Systems
System Workload & Performance
Internal & External Users
User/Customer Usage Profiles
Interoperability Considerations
Security Measures
Operations & Logistics
Legacy System
Legacy Support Environments
Development & Maintenance Environment
Test & Integration Environment
Project Management Environment
Target OperationalEnvironment
TargetCoreSystem
Architecture
User Interface
System Databases
Quality System Attributes
Products & Services
System Documentation
Organization & Customer Sites
Networks (WANs & LANs)
Interfacing Systems
System Workload & Performance
Internal & External Users
User/Customer Usage Profiles
Interoperability Considerations
Security Measures
Operations & Logistics
Target System
Target Support Environments
Prototyping Environment
Development Environment
Integration & Test Environment
Maintenance Environment
Project Management Environment
System Evolution Initiative
Project
MANAGEMENT Activities
Planning and Organizing
Budgeting ~~and Scheduling~~
Tasking and Contracting
Overseeing and Coordinating
Assessing and Resolving
Monitoring and Reporting
INFRASTRUCTURE Support
Project Leader & Project Staff
Project Directives & Guidelines
Project Resources & Assets
Contracts & Procurements
Project Work Products
Project Processes
Key Project Work Products
Project Plan
Deployment & Transition Plan
Configuration Management Plan
Risk Management Plan
Concept of Operations
Target System Requirements
Software Development Plan
Baseline System Documentation
Key Work Products
Strategic Plan
Life-Cycle Model
Statement of Need (SON)
Coordination Plan
Procurement Regulations
Security Regulations
Quality Assurance Plan
Project Charter
Organization
MANAGEMENT Activities
Strategic Business Planning
Marketing and Customer Liaison
Information Technology Planning
Budgeting & Managing Resources
Organizing & Coordinating Projects
Overseeing & Evaluating Projects
Managing Infrastructure Support
INFRASTRUCTURE Support
Organizational Staff
Business Operations
Resources & Corporate Assets
Corporate Agreements & Contracts
Policies & Standards
Contracting & Licensing
MSG, SEPG, and TWGs
Training & Technology Transition
Organization Work Products
Organization Processes
Software Engineering
S/W Requirements Engineering
Legacy Software Analysis
S/W Architecture & Design
Software Development
S/W Test & Integration
S/W Validation & Verification
Software Maintenance
Key Project Processes
Requirements Management
Risk Management
Acquisition Management
Configuration Management
Progress Tracking & Reporting
Issue Resolution
User Training & Support
Systems Engineering
Requirements Engineering
Legacy System Analysis
System Architecture & Design
System Development
System Test & Integration
System Validation & Verification
System Deployment & Transition
System Operations & Maintenance
Technologies
Key Organization Processes
Business Needs Analysis
Startegic Planning
Business Process Reengineering
System Life Cycle Process
Review & Approval Process
Global Issue Resolution
Quality Assurance

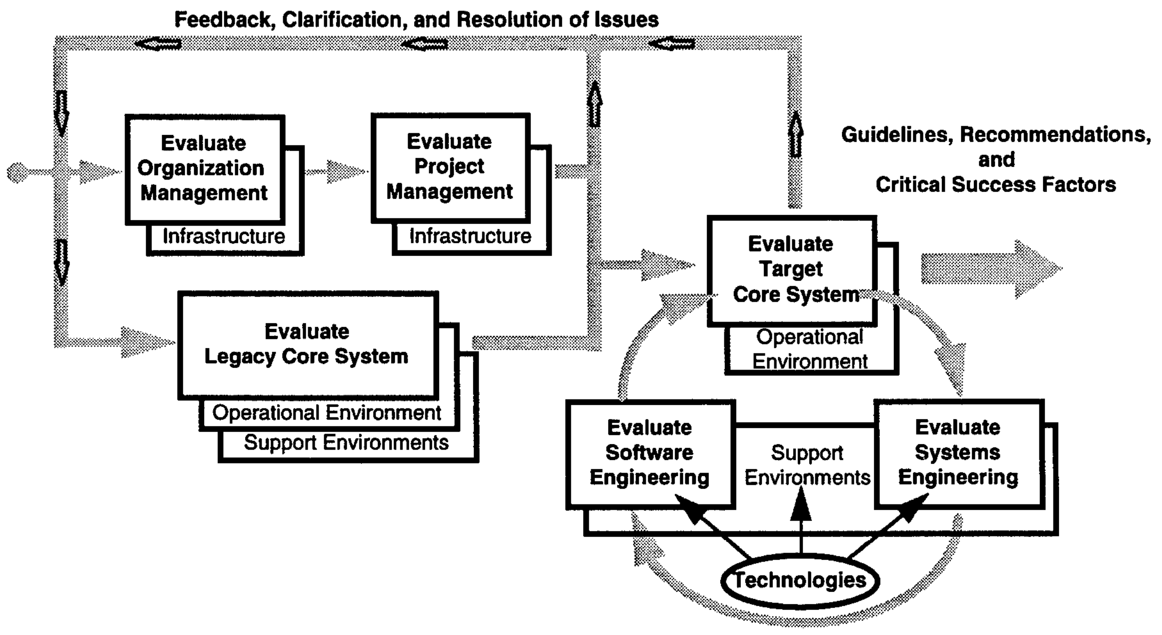
Figure 12: An Expanded View of the Framework Elements

9. How to Use the Framework

Currently we are using the framework for the following types of activities:

- characterizing the problem and solution spaces for system evolution initiatives
- providing guidance to organizations on their strategic and tactical plans for reengineering legacy systems
- identifying technology concerns and potential problems along an organization's projected evolutionary path
- reviewing plans, prioritizing technical and programmatic issues, and recommending improvements to system evolution processes and work products

The general approach we are following in using the framework as a guide to assist us in reengineering planning and evaluation is illustrated in Figure 13.



Feedback, Clarification, and Resolution of Issues
Evaluate Organization Management
Infrastructure
Evaluate Project Management
Infrastructure
Evaluate Legacy Core System
Operational Environment
Support Environments
Guidelines, Recommendations, and Critical Success Factors
Evaluate Target Core System
Operational Environment
Evaluate Software Engineering
Support Environments
Evaluate Systems Engineering
Technologies

Figure 13: An Approach for Using the Enterprise Framework

There are two major parts to the approach: the baseline phase and the evolution phase. The baseline phase focuses on the organization, project, and legacy system. The evolution phase focuses on the target system, systems and software engineering, and technologies used to produce the target system. In other words, the first phase focuses on the problem space, and the second phase focuses on the solution space. An overview of these phases is provided in the following sections.

9.1 Baseline Phase

The first step is to analyze the organization and the project. This includes understanding the motivation and rationale for changing the system and reviewing the global enterprise-level questions identified in Section 8. These analyses can be conducted using the checklists provided for each of these framework elements.

In parallel with the organizational and project analyses, a technical assessment of the legacy system is performed to obtain an understanding of its "current" state. This assessment provides the system baseline necessary to understand the proposed solution approach and the implications of moving to the desired target state. This phase is challenging because of the usual problems of understanding legacy systems. These systems have typically undergone a large number of poorly documented changes through the years. The legacy system checklist is used to surface relevant aspects of the current state.

Several iterations through the cycle, as illustrated by the "feedback loop" in Figure 13, may be necessary to obtain a sufficient understanding. The feedback loop enables the clarification of issues and development of closure on key findings as early as possible. The baseline phase lays the groundwork for conducting the evolution phase.

9.2 Evolution Phase

In the evolution phase, the focus switches from understanding the legacy system to understanding the desired characteristics of the target system, or the "future" state. Evaluating the target system includes considering the technical approach and systems and software engineering practices for evolving the system, the technologies being applied, and the evolution of the operational and support environments.

The target state should be defined in terms of a Concept of Operations, which includes proposed operational usage scenarios and a system specification, which describes the system configuration, desired system features and capabilities, functionality, performance, and other quality features. Also, the target system needs to be understood in terms of the changes that will be required to the current system and the architecture that will support those changes.

The other aspect of this phase includes understanding the proposed technologies and how they are integrated into the systems and software engineering approach. The systems and software engineering activities are key to evaluating and applying new technologies. Technologies introduced in other ways, such as management edict, are symptomatic of an undisciplined and ad hoc approach.

During the evolution phase, the checklists for the target system, software engineering, systems engineering, and technologies elements of the framework are used as a guide.

10. Conclusion

System evolution (and in particular, the reengineering of large systems) is a non-trivial undertaking. One of the major challenges of reengineering is to ensure that the introduction of new capabilities does not adversely affect the current systems operation. This may impose significant constraints on the approach to reengineering a system. The enterprise framework helps to meet these challenges by identifying the contributing factors to consider in software evolution. A manager can use the framework as a guide for identifying the enterprise-wide elements that are critical to success.

In the Introduction, we raised a number of representative questions that managers ask and for which they want "hard answers." While explicit answers to these questions are clearly dependent on the nature of the system being evolved and the particular enterprise (i.e., its culture, organizational structure, and goals and objectives), the framework can provide tangible guidance to assist a manager in developing effective solutions. It does this by providing

- a global frame of reference for answering the question (i.e., an enterprise-wide context)
- insight into contributing factors (i.e., the framework elements)
- insight into related activities, processes, and work products
- a set of checklists for probing the relevant management and technical issues

For example, one of the questions that is raised in the Introduction is "What are the critical success factors of system evolution?" In answer to this question, the management and technical issues (identified in the checklists) that a manager determines are not being addressed (in the migration strategy or project plans) constitute an initial set of critical success factors. A more definitive set of critical success factors will be generated when a manager takes action on the following issues that are raised in the checklists:

- Are there subsidiary plans covering risk management, configuration management, quality assurance, and software development? Have the plans been suitably coordinated?
- Is there a formal process for risk assessment and mitigation? Is it performed regularly or is it a one-time activity?

Another, more difficult, question is "How do we evolve the system without adversely affecting operations?" While there is no explicit answer to exactly how it should be done, the checklists cover these transition issues and allude to possible approaches as shown by the following checklist items:

- Are the transition issues associated with operationally deploying the system being addressed?
- Are all of the legacy system products and services on which the users depend identified?
- Has consideration been given to adopting an incremental implementation approach that is driven by the highest priority risks that have been identified to date?
- Is there a change management strategy for accommodating ongoing software changes to the legacy system that occur during the development of the target system?
- Is there a strategy in place for achieving upward software compatibility?
- How will changes to software interfaces with external systems be coordinated?
- Are all of the external artifacts, system files, and procedures on which the users depend identified?
- What is the plan for "roll out" and "cut-over" to the new system?
- Are programs needed for converting existing data files and databases? Will they automatically make the conversion or will user intervention be required?
- Have the training needs been identified for customers and users of the system?
- Will the operation of the legacy system be sustained to allow adequate time for users to obtain training and fully make the transition to the proposed system?
- In the event of a crisis, to what degree can support be rolled back to the legacy operational environment?

The other three questions that are raised in the Introduction are similarly covered by considering the management and technical issues raised in the checklists.

Developing and fully validating effective management and technical practices for software evolution is a long-range undertaking. While it is not realistic for an organization to think it can develop a "one-size-fits-all" set of practices, it is reasonable to expect that an organization can reach a state where the practices they adapt and use achieve predictable and repeatable results. The enterprise framework represents a starting point for assessing the need for developing a synergistic set of management and technical practices and achieving a disciplined approach to system evolution.

References

[Brown 96]	Brown, Alan; Morris, Ed; & Tilley, Scott. *Assessing the Evolvability of a Legacy System* [online]. Available WWW: <URL: http://www.sei.cmu.edu/technology/reengineering/pubs/white-papers/BrMT96> (1996).
[Foreman 97]	Foreman, J.; Gross, J.; Rosenstein, R.; Fisher, D.; Brune, K. et al. *C4 Software Technology Reference Guide - A Prototype.* (CMU/SEI-97-HB-001, ADA320732). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1997.
[Jensen 83]	Jensen, Michael C. "Organization Theory and Methodology." *The Accounting Review* LVIII, 2 (April 1983): 325-326.
[Paulk 93]	Paulk, Mark; Curtis, Bill; Chrissis, Mary Beth; & Weber, Charles. *Capability Maturity Model for Software (Version 1.1)* (CMU/SEI-93-TR-24, ADA263403). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1993.
[SECMM 95]	Systems Engineering Capability Maturity Model Project. *A Systems Engineering Capability Maturity Model, Version 1.1,* (CMU/SEI-95-MM-003, ADA303318). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1995.
[Smith 97]	Smith, Dennis B.; Muller, Huasi A.; & Tilley, Scott R. *The Year 2000 Problem: Issues and Implications* (CMU/SEI-97-TR-002, ADA325361). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1997.
[Weiderman 97]	Weiderman, Nelson; Northrop, Linda; Smith, Dennis; Tilley, Scott; & Wallnau, Kurt. *Implications of Distributed Object Technology for Reengineering* (CMU/SEI-97-TR-005, ADA326945). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1997.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (LEAVE BLANK)	2. REPORT DATE October 1997	3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Enterprise Framework for the Disciplined Evolution of Legacy Systems		5. FUNDING NUMBERS C — F19628-95-C-0003
6. AUTHOR(S) John K. Bergey, Linda M. Northrop, Dennis B. Smith		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-97-TR-007
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/AXS 5 Eglin Street Hanscom AFB, MA 01731-2116		10. SPONSORING/MONITORING AGENCY REPORT NUMBER ESC-TR-97-007
11. SUPPLEMENTARY NOTES		
12.A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS		12.B DISTRIBUTION CODE

13. **ABSTRACT** (MAXIMUM 200 WORDS)

Many organizations are planning to “migrate” their legacy systems to distributed open system environments or a single product line of systems. Many of these efforts are often less than successful because they concentrate on a narrow set of software issues without fully considering a broader set of enterprise-wide management and technical issues. This report describes an enterprise framework that characterizes the global environment in which system evolution takes place and provides insight into the activities, processes, and work products that shape the disciplined evolution of legacy systems. Exemplary checklists are included to identify critical enterprise issues corresponding to each of the framework’s elements. Preliminary results indicate that the enterprise model is a useful tool for probing and evaluating planned and ongoing system evolution initiatives. The model serves to draw out important global issues early in the planning cycle and provides insight for developing a synergistic set of management and technical practices to achieve a disciplined approach to system evolution.

1. SUBJECT TERMS legacy systems, product line systems, system evolution			15. NUMBER OF PAGES 51
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL